CAAM 520: Computational Science II
Spring 2017 Syllabus

Matthew G. Knepley
3021 Duncan Hall knepley@rice.edu

This course concerns *scientific libraries* as the most effective form of communication for advances in scientific computation. We will learn how to design, implement, test, distribute, and maintain a numerical library for finite element (FEM) computation written in a higher level languages, such as C, Fortran, and Python. Emphasis will be placed solving practical computational problems and providing insight to the user on code accuracy, performance, and tradeoffs. We will also cover basic techniques of algorithm design and implementation, architectural optimization, source management, configuration and build tools, documentation, i/o, and visualization.

Students will learn to write software as a library developer, meaning that they will construct software usable by others. This is a crucial skill for modern computational science, which needs an understandable (modular, hierarchical), extensible, maintainable infrastructure.

1 Course Information

Instructor	Matthew G. Knepley
Class times	1pm to 1:50pm on Monday, Wednesday, & Friday
Location	Duncan Hall 1075

A course overview and grading policy are available in accordance with Rice academic policy.

2 Required and Recommended Reading

Most class time will be occupied by collaborative learning, question and answer time, and group activity. Class notes for CAAM 519 provide background material for this course.

The mathematics textbook to rule them all for FEM is

- The Mathematical Theory of Finite Element Methods by Susanne Brenner and L. Ridgway Scott (Springer) ISBN-10: 0387759336

but some other books which could supplement your reading are

- Numerical Methods for Nonlinear Partial Differential Equations by Sören Bartels (Springer) ISBN-10: 3319137964
- Understanding And Implementing the Finite Element Method by Mark S. Gockenbach (SIAM) ISBN-10: 0898716144
- Finite Elements and Fast Iterative Solvers: with Applications in Incompressible Fluid Dynamics by Howard C. Elman, David J. Silvester, and Andrew J. Wathen (Oxford). ISBN-10: 0198528671

The following books are foundational for the software construction, however we will not follow them in any way, and they are intended for reference and outside reading.

- The C Programming Language (2nd Edition) by Brian W. Kernighan and Dennis M. Ritchie (Prentice Hall) ISBN-10: 0131103628.
- The PETSc Manual, S. Balay, S. Abhyankar, M. Adams, J. Brown, P. Brune, K. Buschelman, V. Eijkhout, W. Gropp, D. Kaushik, M. Knepley, L. Curfman McInnes, K. Rupp, B. Smith, and H. Zhang, ANL-95/11 Rev 3.5, 2015.
- LaTex Companion (Tools and Techniques for Computer Type Setting), 2nd ed., by Frank Mittelbach, Michel Goossens, Johannes Braams, David Carlisle, and Chris Rowley (Addison Wesley Professional) ISBN-10: 0201362996.

3 Schedule

3.1 Week 1: Introduction to FEM 1/9

3.1.1 Lecture 1: Course Introduction

The paper is the perfect vehicle for mathematical communication, but not for communication of results in scientific computing. In this arena, it is the numerical library. I will discuss the creation of software libraries.

I will also the three parts of the course, and the associated programming project for each.

Online resources

- Run-time extensibility and librarization of simulation software, Jed Brown, Matthew G. Knepley, and Barry F. Smith, IEEE CSE, 17(1), 38-45, 2015.
- Short talk on Library development

3.1.2 Lecture 2: Read up on FEM

I am out of town.

3.1.3 Lecture 3: FEM Basics

I go over simple PDE formulations and the finite element method.

Online resources

- From Euler, Ritz, and Galerkin to Modern Computing, Martin Gander, SIAM Review, 54(4), 627–666, 2012.

3.2 Week 2: FEM Tools 1/16

3.2.1 Lecture 4: Holiday: No Class

Read up on FIAT.

Online resources

- Online documentation for FIAT, as part of the FEniCS Project
- Algorithm 839: FIAT, a new paradigm for computing finite element basis functions, ACM TOMS, **30**(4), 502–516, 2004. [Compendia]
- Automated generation and symbolic manipulation of tensor product finite elements, SIAM SISC, **38**(5), S25–S47, 2016. [arXiv]

3.2.2 Lecture 5: FIAT

I introduce the FIAT and FInAT packages and the computational approach to element development.

I describe the first programming project.

3.2.3 Lecture 6: Plex basics

I introduce the Plex framework for mesh representation (see Chapter 7 in CAAM 519 book).

3.3 Week 3: Residuals and Jacobians 1/23

3.3.1 Lecture 7: Residual Calculation I

We look at residual calculation on an element, and pointwise physics functions.

Online resources

- Achieving High Performance with Unified Residual Evaluation

- Finite Element Integration on GPUs
- Finite Element Integration with Quadrature on the GPU
- MOOSE: A parallel computational framework for coupled systems of non-linear equations

3.3.2 Lecture 8: Residual Calculation II

We look at residual calculation over a domain.

3.3.3 Lecture 9: Element Discussion

We discuss progress on the first programming project.

3.4 Week 4: Boundary Conditions 1/30

3.4.1 Lecture 10: Jacobian Calculation

We look at Jacobian calculation over an element.

3.4.2 Lecture 11: Boundary Conditions I

We discuss handling boundary conditions with PetscSection (see Chapter 7.2.2 in the CAAM 519 book).

3.4.3 Lecture 12: Boundary Conditions II

We assemble full solutions and local solutions with boundary values.

3.5 Week 5: Project Presentations 2/6

3.5.1 Lecture 13: Project Presentation I

Students present their first programming project for a new element.

3.5.2 Lecture 14: Project Presentation II

Students present their first programming project for a new element.

3.5.3 Lecture 15: Holiday: No Class

3.6 Week 6: Building a Simulation From Scratch 2/13

3.6.1 Lecture 16: SNES ex12 I

We create SNES ex12 from scratch.

3.6.2 Lecture 17: SNES ex12 II

We create SNES ex12 from scratch.

3.6.3 Lecture 18: SNES ex12 III

We create SNES ex12 from scratch.

3.7 Week 7: Vectorization 2/20

3.7.1 Lecture 19: Vectorization

I introduce vectorization and technologies for enabling it.

I describe the second programming project.

3.7.2 Lecture 20: Vectorizing FEM I

We discuss vectorization of finite element integrals.
Online resources

- Finite Element Integration with Quadrature on the GPU
- OpenCL is an open standard for SIMT programming, used extensively by ViennaCL and PyOpenCL
- CUDA is the SIMT programming system from Nvidia, which is easily used through PyCUDA
- ISPC is a compiler that inserts Intel intrinsics (also look at non-temporal stores)
- LIBXSMM is a library with Intel JIT compilation for small, dense or sparse matrix multiplications, and small convolutions.
- Example PETSc integration using OpenCL

3.7.3 Lecture 21: Vectorizing FEM II

We run some experiments.

3.8 Week 8: Self-Study 2/27

3.8.1 Lecture 22: Conference: No Class

I am out of town.

3.8.2 Lecture 23: Conference: No Class

I am out of town.

3.8.3 Lecture 24: Conference: No Class

I am out of town.

3.9 Week 9: Code Optimization 3/6

3.9.1 Lecture 25: Cache Tiling

I describe cache tiling (also loop tiling) and show code for this in PETSc.
Online resources

- Loopy, an iteration layout engine, from Andreas Klockner
- The STENCIL computation in the Parboil benchmarks from UIUC shows off cache tiling
- Another approach to this is cache oblivious algorithms, originated by Harald Prokop in his MS thesis, where stuff is recursively chopped up until it fits in cache.

3.9.2 Lecture 26: Project Presentation I

Students present their second programming project on vectorization.

3.9.3 Lecture 27: Project Presentation II

Students present their second programming project on vectorization.

3.10 Week 10: Spring Break 3/13

3.10.1 Lecture 28: Holiday: No Class

3.10.2 Lecture 29: Holiday: No Class

3.10.3 Lecture 30: Holiday: No Class

3.11 Week 11: Problem Formualtions 3/20

3.11.1 Lecture 31: Irregular Evaluation

We examine the irregular evaluation needed for particle methods, and look at the design of DMSwarm.

3.11.2 Lecture 32: Multiphysics Formulations

We discuss

- Poisson with a nonlinear boundary condition
- Variable-viscosity Stokes and Navier-Stokes
- Helmholtz
- Magneto-hydrodynamics
- Magama dynamics

3.11.3 Lecture 33: Group Projects

We choose formulations for the third programming project, and divide into groups.

3.12 Week 12: Solver Design 3/27

3.12.1 Lecture 34: Multilevel Preconditioning

We discuss how to design multigrid preconditioners

3.12.2 Lecture 35: Debugging Multigrid

We discuss the paper by Diskin, Thomas, and Mineck on debugging a multigrid implementation.

Online resources

- On Quantitative Analysis Methods for Multigrid Solutions, SIAM SISC, **27**(1), 108–129, 2005.

3.12.3 Lecture 36: Parallel Preconditioning

We discuss parallel considerations when preconditioning.

3.13 Week 13: Quantifying Codes 4/3

3.13.1 Lecture 37: FEM Pitfalls

We discuss the paper review by Dave May illustrating common pitfalls in FEM programming. We also discuss solver issues for group projects.

Online resources

- Interactive comment on "ELEFANT: a user-friendly multipurpose geodynamics code", Solid Earth Discussions, **6**, C877–C893, 2014.

3.13.2 Lecture 38: Simple Profiling

I discuss the PETSc profiling interface.

3.13.3 Lecture 39: Profiling Real Codes

We discuss how to extract actionable information from complex, practical codes, rather than toy problems. Also, we discuss work-precision discrimination.

3.14 Week 14: Group Work 4/10

3.14.1 Lecture 40: Group Project Work

I am out of town.

3.14.2 Lecture 41: Group Project Work

I am out of town.

3.14.3 Lecture 42: Group Project Work

I am out of town.

3.15 Week 15: Final Evaluation 4/17

3.15.1 Lecture 43: Project Presentation I

Student groups present their third programming project.

3.15.2 Lecture 44: Project Presentation II

Student groups present their third programming project.

3.15.3 Lecture 45: Class Evaluation

We discuss class accomplishments and possible improvements.